Today

- Assignment 3
- Synchronization
  - Semaphore
  - Producer-Consumer example
- Questions about quiz 2
Assignment 3

• Questions
  – Due to Nov 18
Semaphores

• Semaphores are another synchronization method, besides Mutex and Cond. Variables

• A semaphore is a special kind of integer: it can be incremented or decremented atomically.
Semaphores

If the value of the semaphore is 0, it cannot be decremented. Threads that try to decrement it when its value is 0 go to sleep

-Increment: int sem_post(sem_t *sem);

-Decrement: int sem_wait(sem_t *sem);
POSIX Semaphores

`sem_t access_lock;
sem_init(&access_lock, 0, 2);
// thread code…
sem_wait(&access_lock);
// critical section…
sem_post(&access_lock);`
#include <semaphore.h>

int sem_init(sem_t *sem, int pshared, unsigned int value);

Link with -lrt or -pthread.

- Initializes the unnamed semaphore at the address pointed to by sem.

- **value** argument specifies the initial value for the semaphore.
- **pshared** indicates whether this semaphore is to be shared between the threads of a process, or between processes.
- If pshared has the value 0, then the semaphore is shared between the threads of a process.
Sample code

• Code semaphore.c

void* func ( void * ptr )
{
    int x = *((int *) ptr);
    sem_wait(&mutex);       // Down semaphore
    // begin critical section
    counter++;               // end critical section
    sem_post(&mutex);        // up semaphore
}
Producer-Consumer Exercise

Create the producer-consumer relationship with some queue size

- The produced item should be the printout “Item produced!”

- The consumption of an item prints out “Item consumed!”
Producer-Consumer Exercise

Use the provided code Pcsem.c as skeleton
Questions?